II. AMENDMENTS TO THE CLAIMS

The following listing of claims replaces all prior versions, and listings, of claims in the application:

1. (Previously Presented) A computer-implemented method for comparing Application Program Interfaces (APIs) between byte code releases, comprising:

receiving source input corresponding to a first release of byte code and target input corresponding to a second release of the byte code;

transforming the source input into a first list that contains class names associated with the first release of byte code, and the target input into a second list containing class names associated with the second release of the byte code;

finding matching class names between the first list and the second list, and loading classes corresponding to the matching class names;

comparing the loaded classes to identify APIs that have been modified between the first release of byte code and the second release of the byte code, wherein an API has not been modified in case that it maintains a same name, parameter order, parameter types and return types in both the first release of the byte code and the second release of the byte code; and

removing the matching class names from the first list and the second list after the comparing, wherein any class names remaining in the first list represent APIs that have been removed for the second release of the byte code, and wherein any class names remaining in the second list represent APIs that have been added for the second release of the byte code.

2. (Previously Presented) The computer-implemented method of claim 1, further comprising outputting a report identifying at least one of the APIs that have been modified, the APIs that have been removed and the APIs that have been added.

3. (Previously Presented) The computer-implemented method of claim 1, wherein the loading step comprises loading at least one Java class of the first release of byte code and at least one class of the second release of the byte code.

4. (Previously Presented) The computer-implemented method of claim 3, further comprising listing methods of the at least one class of the first release of byte code in the first list, and listing methods of the at least one class of the second release of the byte code in the second list.

5. (Previously Presented) The computer-implemented method of claim 4, wherein the comparing step comprises comparing the methods in the first list to the methods in the second list to identify APIs that have been modified between the first release of byte code and the second release of the byte code.

6. (Previously Presented) The computer-implemented method of claim 5, wherein the removing step comprises removing, from the first list and the second list, any methods in the first list that are identical to methods in the second list based on the comparison, wherein any methods remaining in the first list after the removing represent APIs that have been removed for the

second release of the byte code, and wherein any methods remaining in the second list after the removing represent APIs that have been added for the second release of the byte code.

7. (Previously Presented) The computer-implemented method of claim 1, wherein the source input and the target input comprise JAR files.

8. (Previously Presented) The computer-implemented method of claim 1, wherein the source input and the target input comprise a list of classes.

9. (Previously Presented) The computer-implemented method of claim 1, further comprising inputting class paths common to the first release of Java byte code and the second release of the byte code.

10. (Previously Presented) A system for comparing Application Program Interfaces (APIs) between byte code releases, comprising:

a computer device;

an input system for receiving source input corresponding to a first release of byte code and target input corresponding to a second release of the byte code;

a transformation system for transforming the source input into a first list that contains class names associated with the first release of byte code, and the target input into a second list containing class names associated with the second release of the byte code;

a class matching system for finding matching class names between the first list and the second list;

a class comparison system for comparing classes corresponding to the matching class names to identify APIs that have been modified between the first release of byte code and the second release of the byte code, wherein an API has not been modified in case that it maintains a same name, parameter order, parameter types and return types in both the first release of the byte code and the second release of the byte code; and

a removal system for removing the matching class names from the first list and the second list after the comparison, wherein any class names remaining in the first list represent APIs that have been removed for the second release of the byte code, and wherein any class names remaining in the second list represent APIs that have been added for the second release of the byte code.

11. (Original) The system of claim 10, further comprising a report generation system for generating a report identifying at least one of the APIs that have been modified, the APIs that have been removed and the APIs that have been added.

12. (Previously Presented) The system of claim 10, further comprising a class loader for loading at least one class of the first release of byte code and at least one class of the second release of the byte code.

13. (Previously Presented) The system of claim 12, further comprising a method listing system for listing methods of the at least one class of the first release of byte code in the first list, and for listing methods of the at least one class of the second release of the byte code in the second list.

14. (Previously Presented) The system of claim 13, wherein the class comparison system compares the methods in the first list to the methods in the second list to identify APIs that have been modified between the first release of byte code and the second release of the byte code.

15. (Previously Presented) The system of claim 14, wherein the removal system removes, from the first list and the second list, any methods in the first list that are identical to methods in the second list based on the comparison, wherein any methods remaining in the first list after the removing represent APIs that have been removed for the second release of the byte code, and wherein any methods remaining in the second list after the removing represent APIs that have been added for the second release of the byte code.

16. (Original) The system of claim 10, wherein the source input and the target input comprise JAR files.

17. (Original) The system of claim 10, wherein the source input and the target input comprise a list of classes.

18. (Previously Presented) The system of claim 10, wherein the input system further receives class paths common to the first release of byte code and the second release of the byte code.

19. (Previously Presented) A program product stored on a recordable medium for comparing Application Program Interfaces (APIs) between byte code releases, which when executed, comprises:

program code for receiving source input corresponding to a first release of byte code and target input corresponding to a second release of the byte code;

program code for transforming the source input into a first list that contains class names associated with the first release of byte code, and the target input into a second list containing class names associated with the second release of the byte code;

program code for finding matching class names between the first list and the second list;

program code for comparing classes corresponding to the matching class names to identify APIs that have been modified between the first release of byte code and the second release of the byte code, wherein an API has not been modified in case that it maintains a same name, parameter order, parameter types and return types in both the first release of the byte code and the second release of the byte code; and

program code for removing the matching class names from the first list and the second list after the comparison, wherein any class names remaining in the first list represent APIs that have been removed for the second release of the byte code, and wherein any class names remaining in the second list represent APIs that have been added for the second release of the byte code.

20. (Original) The program product of claim 19, further comprising program code for generating a report identifying at least one of the APIs that have been modified, the APIs that have been removed and the APIs that have been added.

21. (Previously Presented) The program product of claim 19, further comprising program code for loading at least one class of the first release of byte code and at least one class of the second release of the byte code.

22. (Previously Presented) The program product of claim 21, further comprising program code for listing methods of the at least one class of the first release of byte code in the first list, and for listing methods of the at least one class of the second release of the byte code in the second list.

23. (Previously Presented) The program product of claim 22, wherein the program code for comparing compares the methods in the first list to the methods in the second list to identify APIs that have been modified between the first release of byte code and the second release of the byte code.

24. (Previously Presented) The program product of claim 23, wherein the program code for removing removes, from the first list and the second list, any methods in the first list that are identical to methods in the second list based on the comparison, wherein any methods remaining in the first list after the removing represent APIs that have been removed for the second release

of the byte code, and wherein any methods remaining in the second list after the removing represent APIs that have been added for the second release of the byte code.

25. (Original) The program product of claim 19, wherein the source input and the target input comprise JAR files.

26. (Original) The program product of claim 19, wherein the source input and the target input comprise a list of classes.

27. (Previously Presented) The program product of claim 19, wherein the program code for receiving further receives class paths common to the first release of byte code and the second release of the byte code.